Delft University of Technology

Master's Thesis

Incorporating Leveled Homomorphic Encryption-based Private Information Retrieval in Federated eID Schemes to Enhance User Privacy

Author:
Kris Shrishak

Supervisors:
Asst. Prof. Dr. Zekeriya Erkin
Remco Schaar MSc.

Thesis submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical Engineering

Cyber Security Group
Department of Intelligent Systems
Delft University of Technology

May 2016

Thesis Committee:

Chair:	Prof. Dr. Jan van den Berg, Head of the Cyber Security Group, TU Delft
University supervisor:	Asst. Prof. Dr. Zekeriya Erkin, Cyber Security Group, TU Delft
Company supervisor:	Remco Schaar MSc., UL Transaction Security
Committee member:	Asst. Prof. Dr. Huijuan Wang, Multimedia Computing Group, TU Delft

"*The multiple human needs and desires that demand privacy among two or more people in the midst of social life must inevitably lead to cryptology wherever men thrive and wherever they write.*"

David Kahn, *The Codebreakers*

Abstract

Numerous services are being offered over the Internet and require identification of users as in face-to-face interactions. To simplify the authentication procedure and reduce the need to manage multiple credentials to access services, Electronic Identification (eID) schemes have been introduced. eID schemes commonly involve many service providers (SPs) which provide services, such as online shopping, social networks, etc. to users and identity providers (IDPs) which verify the identity of users and facilitate the users to authenticate him/herself to SPs. In federated eID schemes, IDPs store identifiable user information (attributes), often with a unique ID, and attest on these attributes to SPs.

In this work we address the privacy concerns of storing user attributes at the IDP which allows the IDP to profile the behaviour and activities of users. We propose to store the attributes in a privacy friendly manner so that they cannot be directly linked to a particular user even if the data is leaked. Then we include an additional step incorporating private information retrieval (PIR) in the usual authentication flow of federated eID scheme so that the IDP can perform its role of authenticating and managing the user's identity without turning into a privacy hotspot. The privacy enhancement offered by our work needs to be accompanied by privacy-friendly authentication, which does not reveal the identity of the user, to be effective. Finally, through a proof-of-concept implementation we show a practical variant of our scheme in which the IDP, with millions of users, partitions its database.

Acknowledgements

This thesis would not have been possible without the help and support of many people. I want to thank Dr. Zekeriya Erkin for supporting and guiding my research by providing valuable inputs during our meetings; and Remco Schaar for providing industry insights and being constantly involved in my research. I am grateful to both of them for their reviews and suggestions to improve this report.

I also want to thank Ph.D. students of Cyber Security Group in TU Delft for the fruitful discussions during my thesis and for their willingness to help when needed. They have been a pleasant company at work and during our social outings. This includes in particular Majid Nateghizad, Gamze Tillem, Chibuike Ugwuoke and Mingxiao Ma.

Finally, I would also like to thank Henrique Di Lorenzo Pires for accepting me as an intern at UL Transaction Security and proposing an interesting topic for the thesis.

Kris Shrishak
Delft, May 2016

Contents

List of Figures

List of Tables

List of Abbreviations

ABC	**A**ttributes-**b**ased **C**redential
CRT	**C**hinese **R**emainder **T**heorem
DSH	**D**oröz-**S**unar-**H**ammouri
eID	**e**lectronic **ID**entitification
EU	**E**uropean **U**nion
FHE	**F**ully **H**omomorphic **E**ncryption
IDP	**ID**entity **P**rovider
IRMA	**I** **R**eveal **M**y **A**ttributes
LHE	**L**eveled **H**omomorphic **E**ncryption
LTV	**L**ópez-Alt **T**romer **V**aikuntanathan
LWE	**L**earning **W**ith **E**rrors
MPC	**M**ulti **P**arty **C**omputation
NTRU	**N**-th degree **TRU**ncated Polynomial Ring
PIR	**P**rivate **I**nformation **R**etrieval
PPCA	**P**olymorphic **P**seudonym **C**ard **A**pplication
RLWE	**R**ing-**L**earning **W**ith **E**rrors
SHE	**S**omewhat **H**omomorphic **E**ncryption
SP	**S**ervice **P**rovider
SVP	**S**hortest **V**ector **P**roblem

Notations

Symbol	Explanation
a	Denotes a real number
$\lceil a \rceil$	Denotes the smallest integer greater than or equal to a
$\lfloor a \rfloor$	Denotes the largest integer less than or equal to a
$[a]_q$	Denotes a mod q
$\lfloor a \rceil_2$	Denotes rounding of a followed by modulo 2
$\|\|a\|\|$	Denotes the Euclidean norm of a
$\|\|a\|\|_\infty$	Denotes the maximum norm of a
$\mathbb{R}^n$	Denotes the real coordinate space of n dimensions
$\mathbb{Z}[x]$	Denotes the set of integer polynomials
R	Denotes the ring of integers
R_q	Denotes the ring of integers modulo q
$\mathbb{F}_q$	Denotes a finite field of order q
$\vec{b}$	Denotes a vector
$\mathbf{B}$	Denotes a matrix
$\Phi_m(x)$	Denotes the m^{th} cyclotomic polynomial for any positive integer m
$\varphi(m)$	Denotes the euler totient function of m
λ	Denotes the security parameter
$\log a$	Denotes logarithm to the base 2.

Chapter 1

Introduction

Governments around the world have had public authentication schemes by issuing Identity documents to identify a person or verify aspects of a person's personal identity. The reliance on these documents extended beyond government services as private businesses also used them for reliable authentication. But the number of services offered online is rapidly growing. From webshops to online banking, from health to online banking to government services such as tax returns. The lack of common electronic identification (eID) mechanism has forced users to maintain multiple credentials for authentication and identification to service providers (SPs), which has caused security and usability issues. For instance, many SPs employ password-based authentication mechanism which has induced users to reuse the same password at multiple service providers or writing them on paper which are significant security concerns [1]. An alternate approach employed by service providers involves the use of hardware security tokens such as smart cards. Here the user is required to carry multiple smart cards and smart card readers to avail different services. The usage of different tokens by each SP to authenticate and identify users has lead to a situation where many countries in the European Union (EU) have either developed an eID scheme or are in the process of developing one so that users can use a single identification scheme in which identity providers (IDPs) validate the identity of users to multiple service providers (SPs) [2][3]. But development of eID schemes has brought privacy concerns to the forefront.

eID schemes can be broadly categorized in terms of the authentication mechanism they use - federated and direct. A federated eID scheme uses an infrastructure in which the user proves who (s)he is to the IDP and the SP trusts the assertion received from the IDP and obtains the required user attributes from it. User attributes may include date of birth, gender and nationality among others. Though the IDP reduces the burden of users by storing the attributes and managing authentication, it becomes a privacy hotspot. In addition to the access to large amount of user data, the IDP can link the activities of the users as well.

Federated schemes such as Dutch eID scheme, Idensys [4], and one of the proposed schemes using polymorphic pseudonyms [5] as well as GOV.UK Verify [6] [7] suffer from these privacy issues.

On the other hand, in an eID scheme such as the German eID scheme [8] and IRMA (I Reveal My Attributes) [9], where the SP performs direct authentication, the user takes more responsibility as (s)he maintains his/her attributes on a token, such as smart card, issued by the IDP during enrolment and proves who (s)he is directly to the SP. The IDP cannot link user activities as user interacts directly with the SP, though the authentication device used in these schemes are often complicated and revocation is known to be challenging.

The primary function of the IDP in an eID scheme is to authenticate the user to SPs rather than storing identifiable user information and linking user activities. If citizen data is stored in a central database, the data must be handled with care. The database should be encrypted so that even if the database is hacked, the hacker does not have direct access to the data. Data breach of the data held by health insurance company Anthem Inc. and Ashley Madison [10] were among the numerous data breaches in 2015 which revealed private information of millions of citizens [11]. Data breaches can cost companies in millions [12][13] and customers may sometimes drag them to court [14]. Companies also lose customers due to data breaches and consequentially lose their trust [15].

Data breaches are not confined to businesses and companies. Government services have had their share of data breaches. These data breaches have a greater potential for damage as they may contain Social Security Numbers. OPM hack in 2015 [16], leak of about 50 million Turkish citizens data in April 2016 [17] and the leak of the voters list in Philippines [18] are some well known examples. Such a trove of data along with public data archives can be used to infer deeper relations about people and their activities. The data obtained from national databases have abundance of identifiable information which can lead to identity theft of the citizens [19]. Finally, the future is uncertain. Information that may appear harmless today, could be used to attack certain sections of the society in the future as was the case during the Holocaust when the population registry in Amsterdam was used to track down Jews[20].

In our literature survey of eID schemes [21], it was observed that eID schemes employing direct authentication schemes are much more privacy friendly than federated eID schemes. Hence we focus our research on improving the privacy of users in federated schemes. Before explaining our research, we succinctly mention what we mean by privacy and why we consider it important.

As the concept of privacy is used in widely varying fields such as technology, law, politics as well as philosophy, it differs depending on the context in which the term is used. In our work we focus on information privacy, which is applied to the collection, storage and sharing of personal information. Other individuals and organisations should not automatically have

access to an individual's data, and where the data is in possession of another party, "the individual must be able to exercise a substantial degree of control over that data and its use [22]."

1.1 Research Objectives

Here we present the main research question that we attempt to answer in this thesis.

Research Question 1. *Can IDP perform the role of authenticating the user to SPs in a federated eID scheme without becoming a privacy hotspot?*

Current focus of research is on the usage of pseudonyms such that IDPs and SPs have different pseudonyms for the same user so that identity of the user is not revealed if they collude [21]. Further, to prevent the IDP from linking user activities in federated eID schemes, non-revealing authentication procedures as used in EU passports and German eID scheme has been proposed to be incorporated into the Dutch eID scheme using Polymorphic Pseudonym Card Application (PPCA) [5]. But what about user attributes? Attributes are often stored with a unique ID and can be misused to perform mass surveillance and profile citizens. This is a serious privacy concern which is partly addressed by Zwattendorfer and Slamanig [23], who use proxy re-encryption [24] based-approach to enhance the privacy of the Austrian eID scheme and allow selective disclosure of user data. Nuñez et al. [25] integrate proxy re-encryption into OpenID protocol but they allow the possibility of self-assertion of encrypted attributes by the IDP. Both these works require the user to generate and provide a re-encryption key to the IDP. In [5], the authors propose to store encrypted attributes at an attribute provider and use pseudonyms to retrieve the encrypted attributes from the attribute provider. But they do not address the problem of the attribute provider recognizing when the same encrypted attributes have been retrieved and thus being able to profile the users based on the combination of user pseudonym, SP and the encrypted data that was retrieved.

Though measures have been taken to improve the privacy of users in eID schemes, these measures focus on eliminating the threats from parties that are not in possession of privacy-sensitive data. In our work we focus on enhancing user privacy by preventing the IDP (which is in possession of privacy-sensitive data) from knowing which attributes in its possession have been accessed by incorporating homomorphic encryption; therefore preventing the IDP from being able to profile users. Thus we narrow our research question to the following:

Research Question 1 A. *Can leveled homomorphic encryption provide a scalable and efficient solution to improve the privacy of users at the IDP by preventing the IDP from knowing which user it is authenticating to an SP?*

To answer this question we consider a scenario where a non-revealing authentication mechanism is used and assume that the IDP is honest-but-curious and analyse the privacy offered by our work with the same yardstick as was used in the literature survey [21] to analysis existing eID schemes. The privacy properties [26] are mentioned as follows for convenience:

1. Anonymity: Users may use a service without disclosing their identity.
2. Pseudonimity: Users may utilise a service by using pseudonyms.
3. Data minimization: Only the required information about the user must be shared in order to prevent misuse.
4. Unlinkability: Users should be able to use resources and services without others being able to link these activities.
5. Unobservability: Users should be able to use services without being observed by others.
6. Transparency: User data should be obtained only when necessary and after user consent.

The first two properties rely on the authentication mechanism. A non-revealing authentication mechanism as in [8] is anonymous and is considered as a necessary prerequisite for our scheme to be effective. The following are the major contributions of our work:

1. We suggest two methods on how attributes could be stored at the IDP in a privacy-friendly manner.
2. We propose to port a leveled homomorphic encryption (LHE) based private information retrieval (PIR) scheme [27] to a 3-party eID context to retrieve user attributes.
3. By adding a step to the usual authentication flow of federated eID schemes, we give users control over the data that is being shared with the SP. Then we modify our scheme so that it can be easily integrated with existing authentication protocols.
4. We provide complexity analysis as well as the results of a proof-of-concept implementation. We implement two scenarios and analyse the performance.

1.2 Thesis Outline

This thesis is organized as follows: In Chapter 2, we provide background information on homomorphic encryption and PIR. In Chapter 3, we present two methods to store attributes

in a privacy friendly manner and explain our proposal to use PIR in a federated eID scheme. In addition, we provide a complexity analysis of the proposal. In Chapter 4, we provide implementation results and the analysis. Finally in Chapter 5, we present our conclusions to the research question and present open issues for future work.

Chapter 2

Background

This chapter lays the foundation for the chapters that follow by providing necessary preliminaries and computational problems. Then homomorphic encryption is briefly introduced and a leveled homomorphic encryption scheme is described. Finally, we describe PIR and a leveled homomorphic encryption-based PIR.

2.1 Preliminaries

Lattices

A lattice $\mathcal{L}$ is a discrete subgroup of $\mathbb{R}^m$, represented by all the integer linear combinations of n linearly independent vectors $\{\vec{b}_1, \ldots, \vec{b}_n\} \in \mathbb{R}^n$ where $m \geq n$. The sequence of vectors $\mathbf{B} = \{\vec{b}_1, \ldots, \vec{b}_n\}$ is called a lattice basis. A lattice may have many different bases.

Noise Distribution

Let χ be a probability distribution on a ring R that samples small elements with high probability. The distribution χ is B-bounded, that is, the magnitude of coefficients of a polynomial sampled from χ is guaranteed to be less than B [28].

2.2 Computational Problems

Shortest Vector Problem

Given a basis $\mathbf{B}$ for a lattice $\mathcal{L}$, find a non-zero vector $\vec{v} \in \mathcal{L}$ such that $||\vec{v}|| = \lambda_1(\mathcal{L}(\mathbf{B}))$, where $\lambda_1(\mathcal{L})$ is the minimum distance of lattice $\mathcal{L}$.

Ring Learning With Errors (RLWE) Problem

The decisional ring learning with error problem first introduced in [29] can be stated as follows. Given polynomially many (a, b), such that $a \leftarrow R_q$, determine if b were generated randomly from R_q or were constructed as $b = a \cdot s + e$ where $s \leftarrow R_q$ is a secret ring element and the error $e \leftarrow \chi$.

Decisional Small Polynomial Ratio (DSPR) Problem

The decisional small polynomial ratio problem first introduced in [28] can be stated as follows. Given $h = [2gf^{-1}]_q$ where, $f = 2u + 1$ and $u, g \leftarrow \chi$, determine h from a uniformly random element of R_q.

2.3 Homomorphic Encryption

Homomorphic encryption is a form of encryption which allows processing of ciphertexts and generate an encrypted result which, when decrypted, matches the result of operations performed on the plaintexts [30]. It can be defined as follows:

Let M be the set of plaintexts and let C be the set of ciphertexts. An encryption scheme is homomorphic if for any given key k the encryption E satisfies

$$\forall m_1, m_2 \in M, \qquad E_k(m_1) \odot E_k(m_2) = E_k(m_1 \diamondsuit m_2) \ , \tag{2.1}$$

for some operators $\diamondsuit$ in M and $\odot$ in C.

If $(M, \diamondsuit)$ and $(C, \odot)$ are two groups, then the Equation 2.1 represents group homomorphism. If $\diamondsuit$ is an addition operator, then the scheme is said to be additively homomorphic and if $\diamondsuit$ is a multiplication operator, then multiplicatively homomorphic. Such an encryption scheme, which allows either addition or multiplication of ciphertexts, is known as partially homomorphic encryption scheme. But it is desired to compute functions involving both addition and multiplication. Consider a function f operating on t messages with $parameters$,

$$f(parameters, (m_1, m_2, \ldots, m_t)) \rightarrow (results) \ ,$$

we would like to compute the function homomorphically

$$f(parameters, E_k(m_1, m_2, \ldots, m_t)) \rightarrow E_k(results).$$

There are three desired properties of such an encryption scheme

1. The party performing homomorphic computations must not learn anything about the data, thus maintaining privacy of the data.

2. The ciphertext size should not grow with the complexity of the function being evaluated. This property is known as ciphertext compactness.

3. The chiphertext expansion should be small.

Though Rivest, Adleman, and Dertouzos proposed the idea of homomorphic cryptosystems in their 1978 paper [31], it was only in 2009 that the first fully homomorphic encryption (FHE) scheme was proposed by Gentry [32]. Gentry's scheme included three major stages - somewhat homomorphic encryption (SHE), Squashing and Bootstrapping. SHE is limited to evaluating low-degree polynomials over encrypted data because after a certain amount of computations too much error accumulates such that the decryption leads to a wrong value. Squashing modifies the decryption circuit of the original SHE scheme to make it bootstrappable while bootstrapping refreshes the ciphertext by homomorphically applying the decryption procedure and obtaining a new ciphertext that encrypts the same value as before but has lower noise. The scheme, based on ideal lattices, was not suitable for practical implementation as it was computationally expensive and the ciphertext sizes were also very large [33]. Variants of the scheme based on different hardness assumptions also turned out to be impractical as the noise contained in the ciphertexts could not be managed after certain number of homomorphic operations and required an expensive bootstrapping step to refresh the ciphertext. But real world applications do not need to be able to handle arbitrary circuits. Thus a leveled homomorphic encryption (LHE) scheme, which is designed to handle a circuit of known depth, is sufficient. Brakerski, Gentry and Vaikuntanathan [34] introduced one such scheme in which the noise grew linearly with multiplicative depth. Another such scheme based on NTRU (N-th degree Truncated Polynomial Ring) [35] was proposed by López-Alt, Tromer and Vaikuntanathan (LTV)[28].

We present the difference between SHE, LHE and FHE in Table 2.1 before presenting LTV scheme.

2.3.1 Leveled Homomorphic Encryption

LTV presented a multi-key homomorphic encryption scheme based on the modified NTRU proposed by Stehlé and Steinfeld [36]. Their work included a LHE scheme which makes use of modulus reduction [34] to manage noise after every multiplication.

TABLE 2.1: Categorization of homomorphic encryption schemes

SHE	No precise bounds on the complexity of the functions that can be evaluated. Ciphertext growth is exponential.
LHE	Circuits of linear multiplicative-depth d can be evaluated. Correctness is guaranteed for circuits of depth $\leq d$. Complexity and overhead depend on d.
FHE	Arbitrary circuits can be evaluated and the complexity is independent of the evaluated function.

The LHE scheme is associated with the following parameters:

- the dimension n,
- the cyclotomic polynomial $\Phi_m(x) = x^n + 1$,
- a number d, which defines the maximum depth of circuit that can be evaluated,
- the moduli values represented by decreasing set of primes $q_0 > q_1 > \cdots > q_d$, and the ring $R_{q_i} = \mathbb{Z}_{q_i}[x]/\Phi_m(x)$ for $i \in [0, \ldots, d]$,
- the noise distribution χ is a truncated Gaussian distribution $D_{\mathbb{Z}^n,\sigma}$ where $\sigma > 0$ is the standard deviation.

Encryption Scheme

A single key version of the LTV scheme as proposed in [37] is described.

- $\texttt{LHE.KeyGen}(1^\lambda)$: For every $i \in [0, \ldots, d]$, sample the polynomials $u^{(i)}, g^{(i)} \leftarrow \chi$. Set $f^{(i)} = 2u^{(i)} + 1$ such that $f^{(i)}$ is invertible in modulo q_i and $h^{(i)} = [2g^{(i)}(f^{(i)})^{-1}]_{q_i}$. Resample $u^{(i)}$ if $f^{(i)}$ is not invertible. Output

$$\texttt{sk} = f^{(i)}, \texttt{pk} = (h^{(0)}, q_0).$$

- $\texttt{LHE.Encrypt}(\texttt{pk}, b)$: To encrypt a message bit $b \in \{0, 1\}$, sample $e, s \leftarrow \chi$ and output the ciphertext

$$c^{(0)} = [b + 2e + h^{(0)}s]_{q_0}.$$

- $\texttt{LHE.Decrypt}(\texttt{sk}, c^{(i)})$: To decrypt a ciphertext $c^{(i)}$, compute

$$b = [[f^{(i)}c^{(i)}]_{q_i}]_2.$$

  The decryption will be correct if $||c^{2^i} f^{2^i}||_\infty < q_i/2$.

- $\texttt{LHE.Add}(c_1^{(0)}, c_2^{(0)})$: If $c_1^{(0)}$ and $c_2^{(0)}$ are encryptions of b_1 and b_2, then output $c_{add}^{(0)} = c_1^{(0)} + c_2^{(0)}$.

- `LHE.Mult`$(c_1^{(0)}, c_2^{(0)})$: Compute $\hat{c}_{mult}^{(0)} = [c_1^{(0)} \cdot c_2^{(0)}]_{\Phi_m(x)}$ and then perform modulus reduction to obtain

$$c_{mult}^{(1)} = \lfloor \frac{q_1}{q_0} \hat{c}_{mult}^{(0)} \rceil_2.$$

  The same process holds for evaluating i^{th} level ciphertext, that is, obtaining $c_{mult}^{(i)}$ from $c_1^{(i-1)}$ and $c_2^{(i-1)}$.

Security Analysis and Parameters

To select safe parameters, the security of LTV scheme needs to be analysed. Here we present a security analysis similar to the description in [37]. The security of LTV scheme can be reduced to the RLWE problem through the following arguments [37]:

1. The hardness of DSPR problem allows to replace $h = [2gf^{-1}]_q$ by some uniformly sampled h'.
2. On replacing h with h' the ciphertext $c = [b + 2e + h's]_q$, which is in the form of the RLWE problem, can be replaced with $c' = [b + u]_q$ with a uniformly sampled $u \leftarrow R_q$, thereby ensuring security.

Stehlé and Steinfeld have shown that the DSPR problem is hard even for unbounded adversaries for their parameter selection ($\sigma > \sqrt{q}\,\mathtt{poly}(n)$). But their parameters are too large to permit even a single homomorphic multiplication [28]. To support homomorphic evaluations, much smaller value of σ is required and hence it is assumed that DSPR problem holds for this choice of parameters[1]. In spite of these assumptions, concrete parameters are hard to choose. The common approach is to assume RLWE follows the same behaviour as the LWE problem [39]. Under this assumption, for λ-bit security, a prime number q and $\sigma = 1$, the dimension is bounded by $n \leq \log(q)(\lambda + 110)/7.2$.

The standard attack on LWE problem requires many samples generated with the same secret s while in LTV scheme, s is randomly generated and independent, thus it is suggested that standard attacks against LWE problems cannot be directly applied to the LTV scheme [37]. However, attacks against the original and modified NTRU can still be applied. Hence an approach similar to the original NTRU paper [35] is used to find the secret f by considering the following 2n dimensional lattice:

[1] A subfield attack on this assumption reducing the security of the encryption scheme has been made available in pre-print [38]. We had progressed deep in our work when this attack was published. As our work uses this encryption scheme as a tool, we believe it can be replaced by another secure LHE scheme.

$$\mathcal{L} = \left(\begin{array}{c|cccc} & h_0 & h_1 & \cdots & h_{n-1} \\ & h_{n-1} & h_0 & \cdots & h_{n-2} \\ \mathbf{I} & \vdots & \vdots & \ddots & \vdots \\ & h_1 & h_2 & \cdots & h_0 \\ \hline \mathbf{0} & \multicolumn{4}{c}{q\mathbf{I}} \end{array}\right)$$

where h_i are the coefficients of $h = 2gf^{-1}$.

$\mathcal{L}$ contains a short vector $v = (f, 2g)$. To see that v is in $\mathcal{L}$, let us consider $u = \frac{-fh+2g}{q}$. Then

$$(f_0, f_1, \ldots, f_{n-1}, u_0, u_1, \ldots, u_{n-1}) \left(\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_{n-1} \\ 0 & 1 & \cdots & 0 & h_{n-1} & h_0 & \cdots & h_{n-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & h_1 & h_2 & \cdots & h_0 \\ \hline 0 & 0 & \cdots & 0 & q & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & q \end{array}\right)$$

$$= (f_0, f_1, \ldots, f_{n-1}, 2g_0, 2g_1, \ldots, 2g_{n-1})$$

Thus we have transformed the problem to search for the shortest vector. To select concrete parameters based on the hardness of shortest vector problem (SVP), Hermite factor (γ) introduced by Gama and Nguyen [40] is defined as

$$\gamma^{2n} = \frac{||\vec{b}_1||}{\texttt{vol}(\mathcal{L})^{1/2n}} \tag{2.2}$$

where $||\vec{b}_1||$ is the length of the shortest vector. In [41] Lindner and Peikert gave experimental results regarding the relation between the Hermite factor and the recovery time as $t(\gamma) := \log(T(\gamma)) = 1.8/\log(\gamma) - 110$. For $n = 1.0066^n$, about 2^{80} seconds on an AMD Opteron running at 2.5 Ghz is needed [41]. For an NTRU lattice, Coppersmith and Shamir [42] showed that it is not needed to find the secret key as most vectors with similar norm will decrypt ciphertexts successfully. Also, an attacker would gain useful information with a lattice vector

as close as norm $q/4$ to the original secret key vector. On setting $||\vec{b}_1|| = q/4$ for $\texttt{vol}(\mathcal{L}) = q^n$, Equation 2.2 becomes

$$\gamma^{2n} = \frac{q/4}{(q^n)^{1/2n}} = \sqrt{q}/4 \tag{2.3}$$

2.4 Private Information Retrieval

To retrieve information from a remote database server, a user is usually required to send queries in the form of indices or keywords. But to preserve the privacy of users, it is required to keep these queries private without crippling the server's ability to retrieve the desired information. Private information retrieval (PIR) provides access privacy to the user by preventing the database administrator from being able to learn which database item was accessed.

Chor et al. [43] showed that in a information theoretic setting, transferring the entire database to the user and having the user retrieve the desired database item locally has optimal communication complexity. Since then solutions have been proposed in which multiple non-colluding servers holding copies of the database provide sub-linear communication complexity [43] [44].

Chor and Gilbao proposed computational PIR (cPIR) in which the difficulty of the server administrator to learn the item accessed by the user is based on computational difficulty [45]. Since then several single-server PIR schemes have been proposed using some intractability assumption [46][47][48][49][50]. But in 2007, Sion and Carbunar concluded that none of the then existing single server cPIR schemes based on number theory were practical as they required one or more modular multiplication per database bit. Since then, lattice-based homomorphic schemes have been developed and the efficiency of their implementation has been improving by two orders of magnitude per year [27]. A cPIR scheme introduced by Doröz-Sunar-Hammouri (DSH) is presented next.

2.4.1 Doröz-Sunar-Hammouri PIR

Doröz-Sunar-Hammouri (DSH) [27] introduced a bandwidth-efficient cPIR scheme using leveled homomorphic encryption with depth d. A database $\texttt{DB}$ with 2^ℓ rows is considered, $\ell = 2^d$, whereby the database can be represented as a shallow binary tree. The indices of the database elements are represented by ℓ bits while the data contained in the row $i \in \{0,1\}^\ell$ is represented as $\texttt{DB}[i]$. A $\texttt{PIR.Setup}$, independent of the queries and the contents of $\texttt{DB}$, is performed before the PIR protocol begins. For a security parameter λ, the PIR protocol including $\texttt{PIR.Setup}$ is described as follows:

- $\texttt{PIR.Setup}(1^\lambda)$: The setup procedure involves the generation of the keys for the homomorphic scheme $(\texttt{sk}, \texttt{pk}) \leftarrow \texttt{LHE.KeyGen}$.
- $\texttt{PIR.Query}(1^\lambda, \texttt{pk}, i)$: The $\texttt{query}$ is generated by encrypting the index i of the item in the $\texttt{DB}$. The index is encrypted bit-wise and then batched before sending to the PIR server.

  $$\texttt{query} \leftarrow \texttt{LHE.Enc}_{\texttt{pk}}(i)$$

  where $\texttt{query} = [\xi_0(x), \xi_1(x), \ldots, \xi_{\ell-1}(x)]$ and $\xi_k(x)$ is the polynomial obtained on the encryption of bit k.
- $\texttt{PIR.Response}(1^\lambda, \texttt{DB}, \texttt{query})$: On receiving $\texttt{query}$, the response is generated by computing:

  $$\texttt{resp} \leftarrow f(\texttt{query}) = \sum_{j \in [2^\ell]} (\texttt{query} = j)\texttt{DB}[j] \pmod 2$$

  As $\texttt{query}$ is the encryption of bits, the function $f(\texttt{query}) = f(\texttt{LHE.Enc}_{\texttt{pk}}(i))$ is computed homomorphically. The bitwise equality test $(\texttt{query} = j)$ is computed as $\prod_{k \in [\ell]}(\xi_k(x) + j_k(x) + 1)$, where $j_k(x)$ is the encoding of bit k of the index j into a polynomial. After testing if each bit of $\texttt{query}$ is equal to the corresponding bit of j, the result is multiplied with the data value $\texttt{DB}[j]$.

  To get an intuitive understanding we explain what happens within the encryptions. If all k bits of $\texttt{query}$ are equal to the corresponding bits of j, the product $\prod_{k \in [\ell]}(\xi_k(x) + j_k(x) + 1)$ gives a 1 and otherwise the product is zero. The equality test will be true for only one index and hence the summation $\sum_{j \in [2^\ell]} (\texttt{query} = j)\texttt{DB}[j] \pmod 2$ will not be affected by the data in other indices. Hence $f(i) = \texttt{DB}[i]$. Through this arithmetic formulation, the database owner does not know which row of the database has been read.
- $\texttt{PIR.Decrypt}(1^\lambda, \texttt{sk}, \texttt{resp})$: On receiving the response, the answer to the query is decrypted as

  $$b \leftarrow \texttt{LHE.Dec}_{\texttt{sk}}(\texttt{resp}).$$

The communication complexity of DSH PIR is $|\texttt{query}| + |\texttt{resp}|$ and involves one round of communication. As ℓ bits are encrypted to form the $|\texttt{query}|$, the size of the $|\texttt{query}|$ is $\ell \cdot \omega$, where ω is the size of a ciphertext. The computation complexity of the scheme includes homomorphic evaluation of $f(i)$. In total the number of homomorphic additions is $\mathcal{O}(2^\ell)$ and homomorphic multiplications per database entry is ℓ and $\ell + 1$ if the database entries are encrypted.

Chapter 3

Design

In this chapter, we present a simple federated eID setting with three parties. Then we present our design, `PReID`, in which two methods to store user attributes at the IDP are suggested in addition to using PIR to retrieve user attributes from the IDP in a 3-party eID context. Subsequently we present modified `PReID`.

3.1 Parties in a Federated eID System

The three essential parties in a federated eID system are as follows:

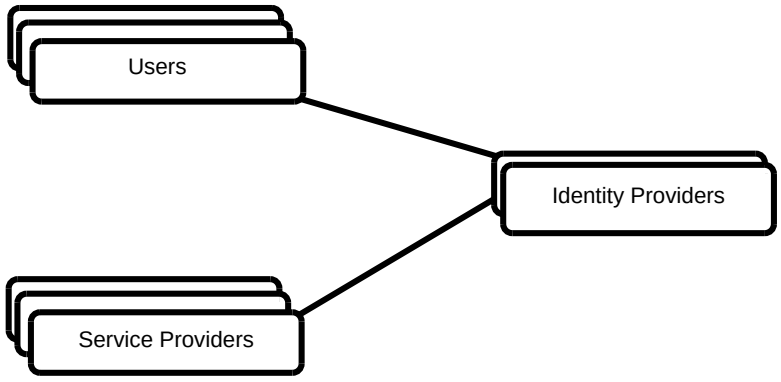

FIGURE 3.1: Parties in a federated eID system

- A *User* is an individual who wants to authenticate her/himself to the Service Provider to access a resource from the SP.

- A *Service Provider* (SP) provides a service, such as online shopping, government tax services, etc. and makes transaction decisions based upon the acceptance of a user's authenticated attributes.
- An *Identity Provider* (IDP) issues and verifies the user's digital identity and facilitates the user to authenticate her/himself to the SP. It improves the overall usability since the user does not need to remember or hold multiple authentication tokens.

The number of users, SPs and IDPs in the system will depend on a number of factors. In a national scheme, the factors include the population of the country, number businesses having operations in the country, etc. In general, millions of users, thousands of SPs and few countable IDPs can be expected to participate in a federated eID scheme. Apart from these three parties, a federated eID system may include additional parties such as brokers. We believe that our idea can be used even if more parties are involved in the scheme as the three primary parties remain the same.

3.2 Federated eID system

The enrolment of users in a Federated eID system is usually performed offline at the IDP, which verifies the user's identity and issues an authentication token which the user can make use of to authenticate to the IDP online. The IDP stores attributes of the user in its database so that they can be sent to SPs after authentication. In Table 3.1, one form of the database is shown. Note that the user attributes at the IDP are stored along with a unique ID. If the authentication token is lost or stolen (in the case of hardware token) or forgotten, revocation is usually performed through a centralized authority or at the IDP.

Table 3.1: Attribute storage

Unique ID	**Date of Birth**	**Name**	**City of Residence**	...
12345678	1/1/1970	Alice	City1	...
14356778	1/2/1970	Bob	City2	...
⋮	⋮	⋮	⋮	...

SPs in the system identify users based on the ability of the user to authenticate to an IDP. Let us consider a situation where a user, say Alice, wants to request a resource from SP_β. She has enrolled in the eID scheme with IDP_α. Then the usual online authentication process of a federated eID scheme with these three parties is described as follows:

1. Alice visits a website, SP_β, which requires her to authenticate in order to get access to the service provided.
2. SP_β displays the list of IDPs whose authentication it accepts.
3. Suppose Alice chooses IDP_α, she is redirected to IDP_α. IDP_α receives a authentication request with a reference to SP_β and the list of requested attributes.
4. IDP_α authenticates Alice.
5. IDP_α sends assertion to SP_β along with the requested attributes of Alice.
6. If the SP_β is satisfied by the authentication, it provides Alice with access to the service she has requested.

In order to prevent linkability in the case of collusion between (1) IDPs and SPs (2) different SPs, transformable pseudonyms have been suggested to be used [5]. But IDPs can still link activities of the user through the pseudonym as it knows which SP receives the authentication assertion and attributes. To address this privacy concern we present `PReID`.

3.3 `PReID`

PIR protocols have been designed for a two-party scenario where the user stores his/her data in a database server and retrieves them when required. We port a two-party protocol to a three-party scenario where the user stores the data with the IDP and sends queries such that the retrieved information is sent to the SP. We call the design of the eID system using PIR as `PReID`.

In this section we present our design `PReID` beginning with a note on the security model and the privacy properties that it offers.

3.3.1 Security Model

We have designed `PReID` for the semi-honest model. The IDP is semi-honest and will try to find out as much as possible about the users, but will not deviate from the protocol. Malicious users and SPs are allowed in the system, that is, users cannot submit queries which they are not permitted to and SPs should not receive attributes that they are not allowed to. If the user deviates from the protocol, it is at a loss as it cannot complete the authentication.

3.3.2 Privacy Requirements

The following privacy requirements are fulfilled by `PReID`:

- IDP does not know which user attributes have been retrieved from its database during the authentication flow.
- The attributes sent from the IDP to the SP is not observable to any third-party.
- IDP and SP cannot link user activities.
- No user can query for the attributes of other users. This requirement is fulfilled by incorporating chip authentication mechanism [8] in which the chip contained in the authentication token issued to the user is authenticated and a secure channel is established between the chip and the IDP.
- SP does not learn user attributes which the user is not willing to share.

3.3.3 Choice of PIR scheme

In an eID scheme, we foresee limited number of computations being performed at the user-side as the token being used might be resource constrained. But PIR schemes require all the involved parties to perform computations. The IDP has greater computation potential than the user while the SP would not want to perform too many computations as their incentive to participate in eID schemes is to eliminate the need for a separate authentication mechanism and its maintenance. Thus we intend to keep the computation performed at the user and SP to a minimum and utilise the computation resources of the IDP. Under these requirements, we chose to use DSH PIR as it is bandwidth-efficient with one round of communication and reduces the computation performed at the user and SP to encryptions and decryptions respectively. For the LHE scheme we considered two schemes - (1) Fan and Vercauteren [51] scheme based on RLWE problem (2) LTV scheme. As their performance is quite similar and as DSH uses LTV scheme in their PIR implementation, we chose the same for ease of comparison. It must be noted that any other LHE scheme can be used in its place and the design of `PReID` is not dependent on LTV scheme.

3.3.4 Attribute Storage

Commonly, user attributes at the IDP are stored along with a unique ID as shown in Table 3.1. Unique ID is used for book keeping purposes and ease of retrieving the attributes after the user is authenticated. Suppose a non-revealing authentication mechanism such as chip authentication mechanism [8] used in German eID scheme is employed, then it is possible to retrieve attributes from the IDP securely, without a unique ID, if the query is sent from an authenticated chip. We propose two methods to store user attributes without the use of unique ID. In addition we also explore the possibility of encrypting the attributes before storing.

Method I

We store user attributes at the IDP at separate database indices, that is, if a user has four attributes, each is stored at a randomly chosen index. For certain types of attributes, this method increases the redundancy as the same attribute value may be stored in multiple locations. For instance, many of the users may be residing in New York City and the attribute 'New York' will be stored in multiple locations. In Table 3.2, it is shown how the attributes from Table 3.1 can be stored when using `PReID`. Note that the attributes of individual users are not grouped together.

Method II

Every attribute value is stored only once at the IDP so that redundancy is avoided. If 1000 users of the scheme reside in New York, then the value 'New York' is stored in only one database entry. The database entries are not filled with redundant values, thus reducing the storage space required by the IDP.

Encrypted Attributes

Instead of storing the attributes in plaintext and trusting the IDP to encrypt the database and safeguard the data, user attributes can encrypted before storing them at the IDP. It is also much more privacy friendly to prevent the IDP from having any kind of access to user attributes after enrolment. As can be observed, only *Method I* can be used for encrypted attributes. The attributes can be encrypted with the public key of SPs or with another public key for which IDP does not have the corresponding private key.

Type 1: If the attributes are encrypted with the public key of SP, then the user can use the services of those SPs with whose public key attributes have been encrypted. This means that, during enrolment, the user should know the SPs whose services (s)he will avail.

Type 2: If the attributes are encrypted with another public key for which neither the user nor the IDP have the private key, then we need to introduce a trusted third-party which will generate the keys for IDP and SP. In addition, a multi-key homomorphic encryption scheme needs to be used.

Our implementation includes plain attributes and *Type 1* of encrypted attributes. Henceforth, we only refer to *Type 1* when we mention encrypted attributes unless explicitly specified otherwise.

3.3.5 Enrolment

Enrolment of the user into the eID scheme is performed offline at the IDP. The user visits enrolment location of the IDP and gets his/her credentials verified. On verification, the

TABLE 3.2: Attribute storage in PReID

Index	Value
1	19700101
⋮	⋮
22	City1
⋮	⋮
41	City2
⋮	⋮
55	19700102
⋮	⋮

IDP stores the necessary attributes of the user in its database using *Method I* or *Method II*. Then the IDP issues a token, which could be in the form of a smart card. Database index numbers, at which user attributes are stored, are saved on the token during the personalization procedure before issuance. In the case of encrypted attributes, additional information such as an identifier for SP and related attributes also needs to be stored on the token. The user can query for attributes stored at the index locations saved on its token. We rely on chip authentication [8] and hence the IDP trusts the chip from which it receives queries. The user has the option of enrolling with more than one IDP. This option allows the user to utilize services offered by SPs in case one of the IDPs is overloaded with requests, as may be the case, for example, during Black Friday, and cannot handle user's queries. But enrolling at too many IDPs voids one of the purposes of eID scheme - reducing the need for multiple credentials.

When a SP joins the eID scheme, it generates a key pair $(\mathtt{sk}, \mathtt{pk}) \leftarrow \mathtt{LHE.KeyGen}$. It selects the IDPs whose authentication it will rely on and shares its public key $\mathtt{pk}$ with the selected IDPs. The IDPs will need the public key to process the queries sent by the user to retrieve the attributes.

3.3.6 Authentication

To generate queries to retrieve the attributes from IDP_α, the user needs to encrypt the indices saved on its token with SP_β's public key, $\mathtt{pk}_\beta$. SP_β can send $\mathtt{pk}_\beta$ to Alice when it displays the list of IDPs whose authentication it accepts.

The online authentication process is shown in Figure 3.2 is described as follows:

1. Alice visits a website which requires her to authenticate in order to get access to the service provided. For instance, if Alice visits a webshop to buy alcohol, then the webshop

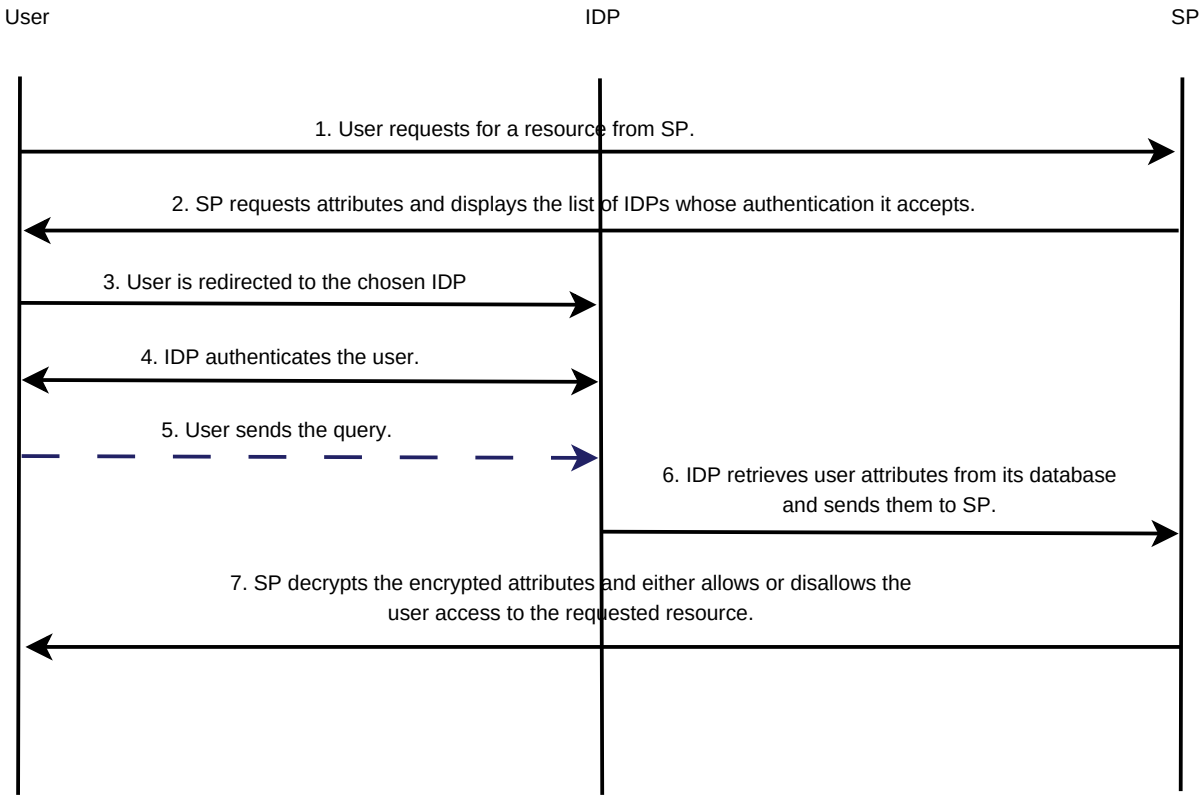

FIGURE 3.2: `PReID` authentication flow

might require her to prove that she is above the minimum age in the country of residence to buy alcohol.

2. SP_β requests an attribute, say Date of birth (DOB), and sends its public key $\texttt{pk}_\beta$ to Alice. It displays the list of IDPs whose authentication it accepts.

3. Alice selects one of the IDPs on the list, say IDP_α, and is redirected to IDP_α. IDP_α receives a authentication request with a reference to SP_β.

4. IDP_α authenticates Alice's token.

5. Once authenticated, Alice performs `PIR.Query` by encrypting the index associated with the attributes requested by SP_β using $\texttt{pk}_\beta$ to generate a `query` and sends it to IDP_α. This step is shown with a dashed line in Figure 3.2.

6. IDP_α performs `PIR.Response` to privately retrieve the attribute. The result of `PIR.Response` is in encrypted form. Thus the retrieved attribute is in encrypted form even if it wasn't stored encrypted. After retrieving the attribute, IDP_α sends it to SP_β.

7. SP_β deciphers the response received from IDP_α by performing `PIR.Decrypt` using $\texttt{sk}_\beta$ and allows or disallows Alice to buy alcohol.

As can be seen, we have been able to incorporate PIR into eID authentication flow with one additional step and prevent IDPs from knowing which attributes have been sent to SPs. On clubbing this process with non-revealing authentication in Step 4, we prevent the IDP from linking user activities. By using a probabilistic LHE in PIR, encrypted attributes are not observable.

The direct transfer of data between SP_β and Alice in the form of $\texttt{pk}_\beta$ leaves open the possibility of phishing attacks. But such an attack is annulled by the fact that the receiver of encrypted attributes will not be able to obtain the attributes on decryption if IDP_α does not have the same $\texttt{pk}_\beta$ as that used by the user to encrypt the query.

But this elegant method is not supported in existing authentication protocols used in federated schemes. Direct transfer of data from SP_β to Alice when SP_β sends its public key and requests for attributes requires modification to the authentication protocol. To integrate with standard protocols such as Security Assertion Markup Language (SAML) and OpenID Connect, we propose a modified version of `PReID` next.

3.4 Modified `PReID`

To integrate `PReID` with existing authentication protocols, we add an additional step after IDP_α authenticates Alice. We are not hiding SP_β from IDP_α, which receives $\texttt{pk}_\beta$ when SP_β joins the eID scheme. So IDP_α can send $\texttt{pk}_\beta$ to Alice once Alice is authenticated. We show the modified authentication flow in Figure 3.3 and describe it below:

1. Alice requests a resource from SP_β.
2. SP_β displays the list of IDPs whose authentication it accepts.
3. Alice is redirected to IDP_α. IDP_α receives a authentication request with a reference to SP_β and the list of requested attributes.
4. IDP_α authenticates Alice's token.
5. IDP_α checks if SP_β is allowed to request for the mentioned attributes. If SP_β has the permissions, then IDP_α sends the list of attributes requested by SP_β to Alice. In addition, IDP_α also sends $\texttt{pk}_\beta$ which Alice can use to encrypt queries. This step is shown with a dotted line in Figure 3.3.

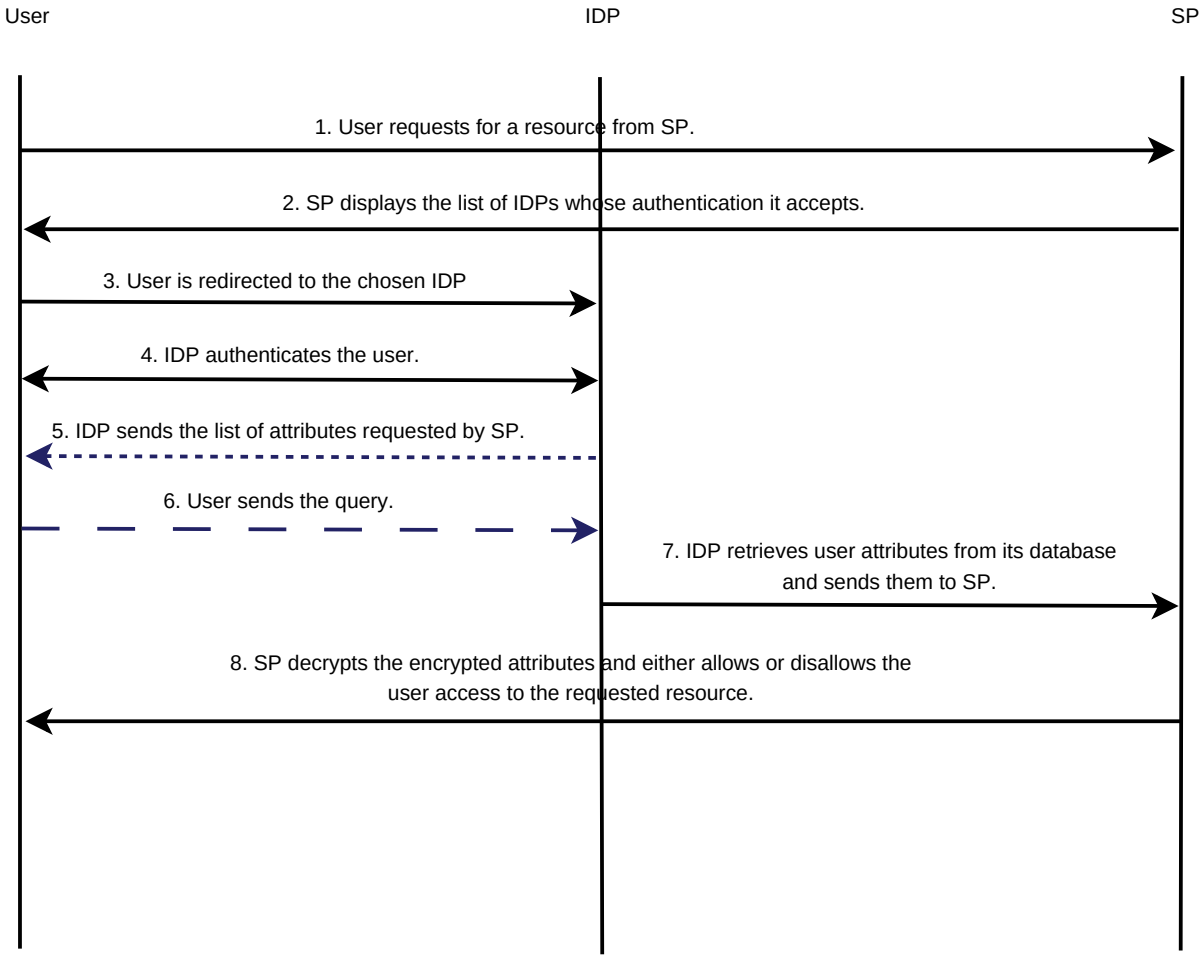

FIGURE 3.3: Modified `PReID` authentication flow

6. Alice performs `PIR.Query` by encrypting the index associated with the attributes requested by SP_β using $\texttt{pk}_\beta$ to generate a `query` and sends it to IDP_α.

7. IDP_α performs `PIR.Response` to privately retrieve the attributes and sends them to SP_β.

8. SP_β deciphers the response received from IDP_α by performing `PIR.Decrypt` using $\texttt{sk}_\beta$ and allows or disallows Alice the access to the requested resource.

In step 3 of the authentication flow of modified `PReID`, SPs send the list of attributes. Here, IDPs get the knowledge of the attributes that each SP requests. But the IDP does not know which user is being authenticated and which index in its database is read to send attribute value. Hence this additional information cannot be exploited by the IDP for malicious reasons. However, this information can be used by the IDP to double-check if the SP has the permission

to request for certain attributes. For instance, a social media website does not need to know user's city of birth and the IDP can prevent it from asking such attributes. This way the IDP plays a role of minimizing the data shared with the SP.

3.5 Complexity Analysis

`PReID` as well as modified `PReID` add the same computation and communication cost to the usual authentication flow in a federated eID system. Though our design can be used with any PIR protocol and LHE scheme, we present the complexity analysis based on DSH PIR and LTV scheme as we use them in our implementation. The communication complexity is limited to the communication complexity of the PIR protocol used, that is $|\texttt{query}| + |\texttt{resp}|$ and the size of the public key $\texttt{pk}_\beta$.

Table 3.3: Computation Complexity

Parties	Operations			
	Encryptions	Decryptions	Additions	Multiplications
User	$\mathcal{O}(\ell)$	-	-	-
IDP	-	-	$\mathcal{O}(2^\ell)$	$\mathcal{O}(\ell \cdot 2^\ell)$
SP	-	$\mathcal{O}(\ell)$	-	-

Computation complexity depends on the PIR protocol where the user performs `PIR.Query`, IDP runs `PIR.Response` and SP runs `PIR.Decrypt`. The user performs ℓ encryptions, where ℓ is the number of bits used to represent a database index. The IDP runs `PIR.Response`, which involves homomorphic additions and multiplications, to retrieve user attributes and sends them to the SP, which then decrypts the response from the IDP. Table 3.3 shows the computation complexity of `PReID`.

Chapter 4

Performance Analysis

In this chapter, we present the details of the proof-of-concept implementation of our design, the choice of parameters and optimizations. We analyse the results and provide few insights on the implications of using our design in a real-world scenario.

To analyse the performance of our design, we have implemented our prototype in C++ on an Intel Core i5-2520M @ 2.5 GHz machine. Through our implementation we show the computation and communication overhead incurred on incorporating PIR in the authentication flow of a federated eID scheme. As the transaction times of various existing federated eID schemes may differ, for generality, we have implemented steps 5-7 of `PReID`. For the single-key LTV scheme we use the C++ implementation of Doröz, Hu and Sunar [37] which relies on NTL [52] and GMP [53] libraries. We have used the NTL library version 9.6 without thread boosting for lattice operations and compiled with GMP 6.1.0.

4.1 Optimizations

4.1.1 Batching

Smart and Vercauteren [54] introduced batching, which allows to evaluate a circuit on multiple independent data inputs simultaneously by embedding them into the same ciphertext. LTV scheme allows the encryption of binary polynomials as messages and hence message bits are encoded to form binary polynomials using Chinese Remainder Theorem (CRT) on a cyclotomic polynomial $\Phi_m(x)$ of degree $n = \varphi(m)$. $\Phi_m(x)$ is of the form

$$\Phi_m(x) = \prod_{i \in [\eta]} F_i(x) \ , \tag{4.1}$$

where η is the number of factors irreducible in $\mathbb{F}_2$ and $\deg(F_i(x)) = k = n/\eta$. The parameter k is the smallest value satisfying $2^k \equiv 1 \pmod{m}$. The factors $F_i(x)$ define message slots in which the message can be embedded. Given a message bit vector $\vec{b} = [b_0, b_1, \ldots, b_{\ell-1}]$, the message polynomial is computed by inverse CRT $b(x) = \text{CRT}^{-1}(\vec{b})$. Batched messages can be extracted by performing modular reduction $b_i = b(x) \pmod{F_i(x)}$. Thus multiplication and addition can be performed: $b_i \cdot b'_i = b(x) \cdot b'(x) \pmod{F_i(x)}$ and $b_i \oplus b'_i = b(x) + b'(x) \pmod{F_i(x)}$.

4.1.2 Reducing Public Key Size

In Section 2.3.1, we mentioned that decreasing set of primes $q_0 > q_1 > \cdots > q_d$ are used as moduli to cope with the growth of noise on homomorphic multiplication. As the modulus is part of the public key, using a set of moduli produces a large public key. A large public key is not beneficial to our cause as it is sent to the IDP when the SP joins the eID system and the IDP sends it to the user during the authentication process. To reduce the public key size, we use an optimization proposed in [37]. We use $q_i = p^{d-i}$ for $i = 0, 1, \cdots, d-1$ where $p \in \mathbb{Z}$.

4.2 Choice of Parameters

In this section, we mention the parameters that we chose for our implementation of `PReID`. For a database of size 2^ℓ, we require a minimum circuit depth $d = \lceil \log \ell \rceil$. We use two circuit depths - $d = 5$ and $d = 4$. In an eID scheme with millions of users, the number of attributes stored in the database will also be in millions. For a realistic database size, $d = 5$ is sufficient as it can handle up to 2^{32} entries. We also consider $d = 4$, which can handle a smaller database of size 2^{16}, to show a comparison and further analysis.

As explained in Section 2.3.1, the parameters n and q_0 need to be such that the Hermite factor $\gamma < 1.0066$ based on Equation 4.2 for 80-bit security. We show the hermite factors we computed in Table 4.1.

$$\gamma^{2n} = \frac{q/4}{(q^n)^{1/2n}} = \sqrt{q}/4. \tag{4.2}$$

In Table 4.1, $(n, \log q_0)$ combinations for which hermite factor is less than 1.0066 is safe for the encryption scheme. Though smaller values of $\log q_0$ will be secure, we cannot use them as we need to cope with noise growth due to homomorphic multiplication. The decryption will be correct if $||c^{2^i} f^{2^i}||_\infty < q_i/2$ and we need to consider that the modulus is reduced at each level of the circuit.

TABLE 4.1: Hermite factor for various q_0 and n values

n	$\log q_0$		
	250	512	640
2^{12}	1.01046	1.02172	1.02727
2^{13}	**1.00522**	1.01080	1.01354
2^{14}	**1.00260**	**1.00539**	1.00675

To reduce the communication complexity, we make use of batching when queries are generated. In [27], DSH call this method bundled queries. The ciphertext size (ω) without batching is $n \log q_0$. With batching, we use η message slots and ω becomes $(n \log q_0)/\eta$. The choice of parameters is summarized in Table 4.2. Our choice of parameters is in line with those chosen in [27].

TABLE 4.2: Chosen parameters

d	m	n	η	$\log q_0$
5	21845	16384	1024	512
4	8191	8190	630	250

4.3 Experimental Results

We have implemented by storing plain attributes of size 8KB as well as encrypted attributes. We consider database size 2^ℓ for which $d = \lceil \log \ell \rceil$ if plain attributes are stored and $d = \lceil \log \ell \rceil + 1$ in case encrypted attributes are stored. Additional depth is required to accommodate the extra homomorphic multiplication in `PIR.Response`. We present `PIR.Response` equation from Section 2.4.1.

$$f(\texttt{query}) = \sum_{j \in [2^\ell]} \Big(\prod_{k \in [\ell]} (\xi_k(x) + j_k(x) + 1) \Big) \texttt{DB}[j] \pmod 2. \tag{4.3}$$

When database entries $\texttt{DB}[j]$ are plain attributes, for each entry, l homomorphic multiplications are performed for bitwise-comparison. If $\texttt{DB}[j]$ is encrypted, then the multiplication with the comparison result is also homomorphic. Thus requiring additional depth.

4.3.1 Query Generation

For the parameters in Table 4.2, the time taken to generate a query and the size of the query are presented in Table 4.3 and Table 4.4. It can be observed that for $d = 5$, encryption

of a single bit takes 515 milliseconds while for $d = 4$, it takes 106 milliseconds. The size of the query for maximum database size of 2^{16} can be compared in Table 4.3 and Table 4.4. Query size is larger for encrypted attributes due to the parameter choices. For encrypted attributes $d = 5$ is required while for plain attributes $d = 4$ is sufficient.

TABLE 4.3: Query size and query generation time to retrieve attributes

Maximum Database Size 2^{ℓ}	Message Slots η	Ciphertext Size ω	Query Size $\ell \cdot \omega$	Query generation time
2^{32}	1024	1 KB	32 KB	16.5 s
2^{16}	630	406.25 B	6.35 KB	1.7 s

TABLE 4.4: Query size and query generation time to retrieve encrypted attributes

Maximum Database Size 2^{ℓ}	Message Slots η	Ciphertext Size ω	Query Size $\ell \cdot \omega$	Query generation time
2^{16}	1024	1 KB	16 KB	8.25 s
2^{8}	630	406.25 B	3250 B	0.9 s

4.3.2 Attribute Retrieval

Attribute retrieval or `PIR.Response` is the most computationally expensive part of `PReID`. Our implementation involves storing attributes with and without encryption.

For plain attributes, we have implemented using a database of maximum 65536 entries for $d = 4$ and until 2 million entries for $d = 5$. We present the results in Table 4.5 and Table 4.6. It can be observed that the retrieval time for database size until 512 is constant for $d = 4$ while it is constant until 1024 for $d = 5$ (Figure 4.1). This is because we batch the database during retrieval for optimization. Batching is performed using $\eta = 8190/13 = 630$ for $d = 4$; $\eta = 16384/16 = 1024$ for $d = 5$. We are able to batch the database as we have stored integers as attributes. So the retrieval is performed block-wise where each block accommodates η message slots. For $d = 5$, a database with 4096 entries uses 4 blocks and a database with 32768 entries uses 32 blocks. Thus, data retrieval time increases linearly with the number of blocks as shown in Figure 4.2. Linearity is accounted for by the increase in the number of homomorphic multiplications performed as the number of blocks increases.

To retrieve encrypted attributes, we have implemented using a database of 256 entries for $d = 4$ and 1024 entries for $d = 5$. Though $d = 5$ allows us to store up to 65536 attributes, we have not implemented beyond 1024 entries as the time taken to retrieve an attribute from database with 1024 entries is about 90 minutes. We present the results in Table 4.7 and Table 4.8. As the attributes are encrypted during enrolment and are stored as a polynomial,

TABLE 4.5: Private retrieval of attributes for $d = 4$

Database Size	Time (s)
16	0.73
32	0.73
64	0.73
128	0.73
256	0.73
512	0.73
1K	1.47
2K	3
4K	5.24
8K	10.5
16K	20.4
32K	40.9
64K	79.8

TABLE 4.6: Private retrieval of attributes for $d = 5$

Database Size	Time (s)
16	6.6
32	6.6
64	6.6
128	6.6
256	6.6
512	6.6
1K	6.6
2K	13.23
4K	26.54
8K	53.21
16K	106.6
32K	213.8
64K	430.87
128K	865.15
256K	1737.46
512K	3497.52
1M	6768.4
2M	13626.5

TABLE 4.7: Private retrieval of encrypted attributes for $d = 4$

Database Size	Time (s)
2	0.95
4	1.91
8	3.85
16	7.67
32	15.46
64	31.8
128	63.9
256	126.8

TABLE 4.8: Private retrieval of encrypted attributes for $d = 5$

Database Size	Time (s)
2	10.5
4	20.7
8	40.8
16	81.8
32	164.5
64	325.1
128	649.4
256	1313.5
512	2630.0
1K	5300.3

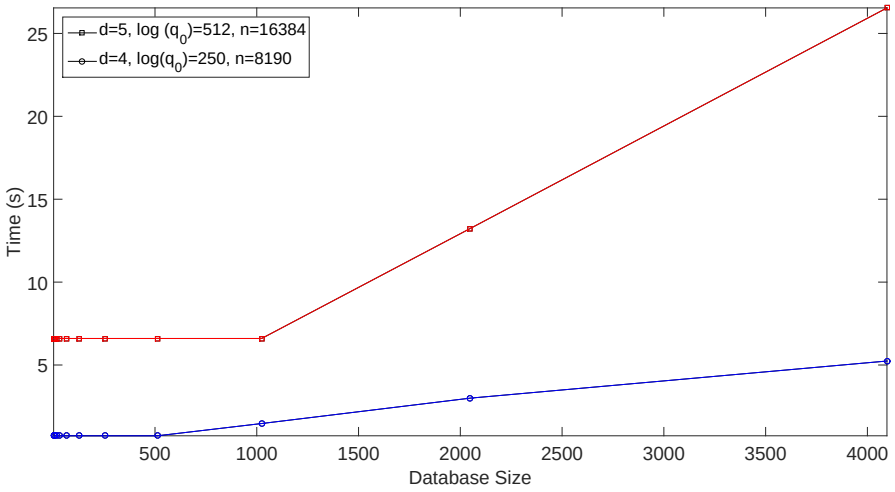

FIGURE 4.1: Comparison of retrieval time for small database size

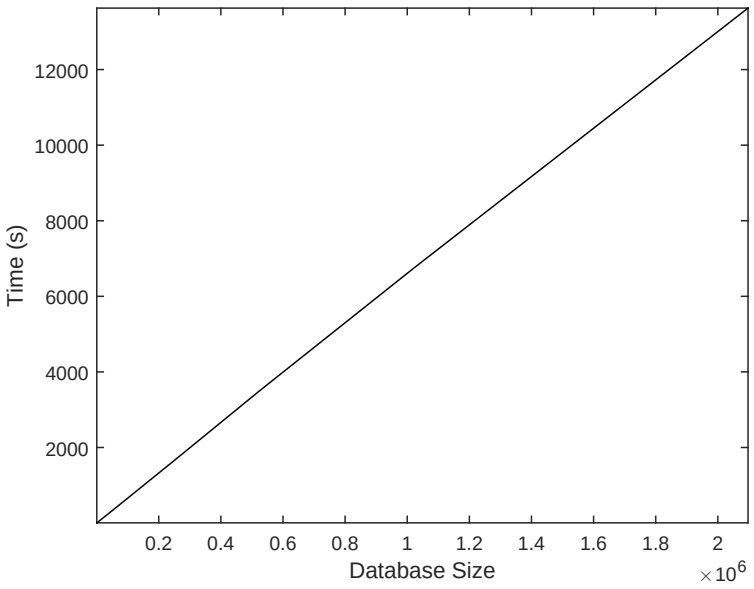

FIGURE 4.2: Retrieval time for $d = 5$

we cannot use batching technique during retrieval. Thus the retrieval time increases linearly with the database size.

4.3.3 Decryption of Attributes

As the message bits are encoded as polynomials before encryption, decryption results in a polynomial which needs to be decoded. The decryption time is constant while the decoding

time depends on the data. For instance, less time is taken to decode if the encoded data obtained after decryption has many zero coefficients. In our experiments, decryption time is 1 second and 12 seconds for $d = 4$ and $d = 5$ respectively.

4.4 Analysis of Results

In this section we analyse the experimental results in terms of efficiency and scalability. We begin by stating that storing attributes in encrypted form at the IDP is privacy friendly as it prevents the IDP from accessing the attributes during storage. But it requires large storage space. If an attribute can be represented in 20 bits, then on encryption for $d = 4$, the ciphertext size is 8125 bytes. The retrieval time for encrypted attributes is much greater than plain attributes. Ideally we will like to store all the encrypted attributes in a single database so that the IDP cannot profile the users. But in terms of efficiency, using a database of maximum size 256 with $d = 4$ gives reasonable performance as encrypted attribute is retrieved in about 2 minutes. Though 2 minutes is not real-time and the user needs to wait, with improvements mentioned in Section 4.5, the duration of wait can be brought down. As eID schemes have millions of users, the IDP can partition its database into multiple small database.

For plain attributes, the difference in retrieval times for $d = 5$ and $d = 4$ is shown in Figure 4.3. Again, the ideal scenario is to use $d = 5$ as it allows up to 4 billion attributes to be stored. But the scheme is not efficient. For efficiency, partitioning the database is suggested. As we consider a binary circuit, we partition the database such that $d = 4$ can be used. Partitioning the database into segments of 65536 entries improves the efficiency as well as makes the solution scalable to support multiple queries in parallel. For $d = 4$ with a maximum of 65536 attributes stored in a partition, attributes are retrieved in about 80 seconds (Table 4.5).

As partitioning of database tables is supported by most database management systems, it is not an inconvenience for the IDP. But in order to retrieve attributes, the IDP requires to know which partition to search. Thus additional information needs to be provided with the query. This mechanism allows for certain level of profiling as the IDP will know the database segment from which attributes were retrieved. If the total number of attributes stored at an IDP is 2 million and these attributes are stored in segments of 65536, then the profiling ability of the IDP increases by 32 times. The privacy of users is still better than it would be in existing federated eID schemes as the probability of identifying which attribute has been retrieved is 1/65536. The total overhead due to `PReID` is shown in Figure 4.4. As mentioned earlier, attribute retrieval time at the IDP increases linearly with the database size while query generation by user and decryption of attributes by SP is constant time.

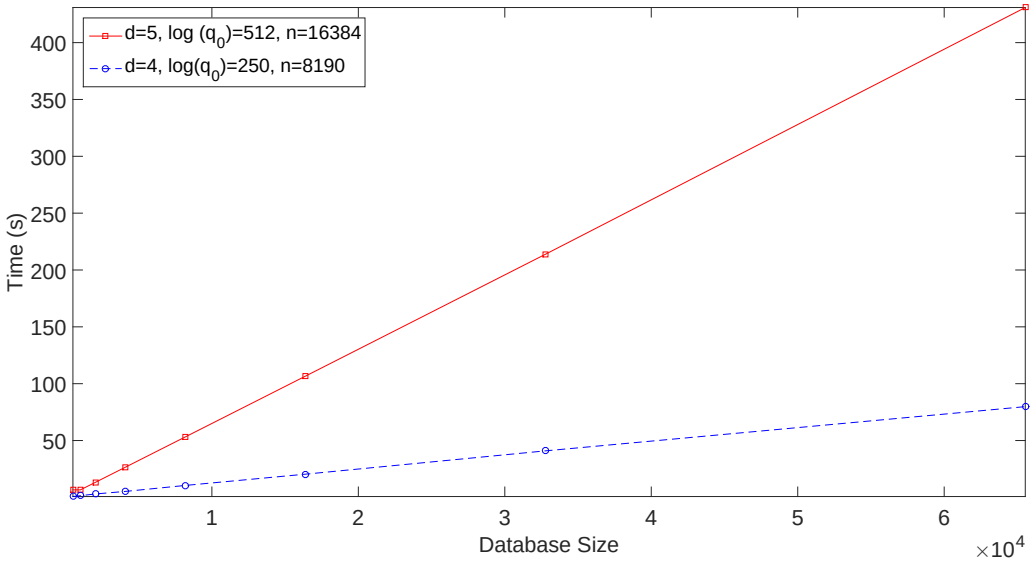

FIGURE 4.3: Comparison of retrieval times for $d = 5$ and $d = 4$

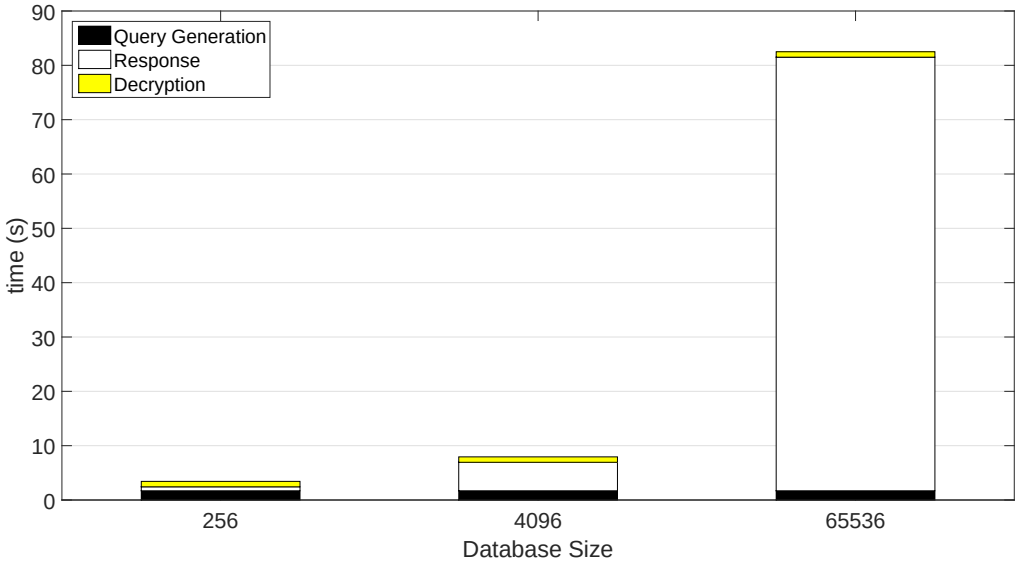

FIGURE 4.4: Total overhead due to `PReID`

4.5 Improvements

We have implemented `PReID` on a 2.5GHz machine. Though we believe that 80 seconds overhead is reasonable in exchange for better privacy, there are possible improvements which could bring the time down. The original implementation of DSH PIR has better performance than ours as they use a 3.5GHz machine. For `PIR.Response`, the IDP would have a much more powerful machine at its disposal. If implemented using multiple cores, the time taken to retrieve data will decrease. In [55], the authors implement DSH PIR using graphics processing units (GPUs). They exploit parallelism in GPUs to provide upto 33 times speedup in `PIR.Response` timing. Such a speedup will allow `PIR.Response` to be run in less than 3 seconds which will make `PReID` more acceptable.

4.6 Practical Implications

We conclude this chapter by considering Trusted Execution Environment (TEE) on mobile phones as a possible deployment environment to store database index numbers and to execute `PIR.Query`. TEE is an environment that allows secure execution of applications. There are several methods to implement TEE, but there is no uniform solution. Thus we summarize the desired security features of TEE provided by Vasudevan et al. [56] as follows:

- Isolated Execution - ensures that applications are executed in isolation from other applications while maintaining the integrity and secrecy of the code and data at run-time.
- Secure Storage - protects stored data belonging to a certain application from being accessed by other applications.
- Remote Attestation - enables remote entities to verify that they are dealing with a particular trusted application on the TEE.
- Secure Provisioning - allows remote entities to send data to a specific application on a specific TEE.
- Trusted Path - protects the authenticity of the communication between a application and a peripheral such as keyboard.

The security features of TEE makes it an attractive alternative to smart cards. TEE can also address the drawback of limited Random Access Memory (RAM) and storage space in smart cards. Mobile phones have more memory than smart cards and are more suitable for lattice-based cryptography than smart cards as public key sizes are large (in megabytes). In spite of this benefit, there remains one concern that requires further research. We draw samples from truncated discrete Gaussian distribution to perform encryption during `PIR.Query`.

Dwarakanath and Galbraith [57] state that implementation of sampling from Gaussian distributions on resource constraint devices may not be practical. We are not sure if this assessment extends to mobile phones as well.

Chapter 5

Discussion and Future Work

In this chapter we discuss the answer to our research question and provide directions for future work. We begin by stating our broad research question.

Research Question 1. *Can IDP perform the role of authenticating the user to SPs in a federated eID scheme without becoming a privacy hotspot?*

We answer this question by stating assertively that IDP can perform its role of authenticating the user to SPs in federated eID schemes without becoming a privacy hotspot. We introduced `PReID` and modified `PReID`. With `PReID`, we suggest methods to store user attributes at the IDP and prevent the IDP from knowing which attributes have been retrieved during the authentication flow. First, storing user attributes at the IDP without a unique ID. Unique ID is commonly used in eID schemes for book keeping purposes and easy retrieval of attributes. However, the use of unique ID makes it easy for IDPs to track the activities of users. Thus we suggested two methods of storing user attributes in a privacy friendly manner. *Method I* involves storing each attribute as a separate database entry. We suggested that attributes of individual users should be stored in non-sequential randomly chosen index. *Method II* reduces the redundancy of data in the database by allowing the IDP to store common attributes such as the city of residency once. In addition to these two methods, we also suggest to encrypt attributes during user enrolment before storing them at the IDP so that the IDP does not have access to the attributes. Storing encrypted attributes requires more space due to ciphertext expansion. In case the encryption is performed using the public key of SP, at the time of enrolment, the user needs to know the services it will avail.

Second, we incorporate PIR into federated eID schemes. PIR has been designed for two-party scenario where the user stores data at a server and retrieves it privately. We port PIR into a three-party scenario where the user queries for the attribute after authentication and

the IDP sends requested attributes to the SP. Using PIR in a federated eID scheme requires an additional step in the usual authentication flow. This step provides users control over which attributes are sent to the SP. `PReID` provides unlinkability and unobservability of attributes while relying on non-revealing authentication mechanism based on chip authentication to provide anonymity.

Modified `PReID` makes it easier to integrate `PReID` into existing authentication protocols. It maintains the privacy properties of `PReID` but allows the IDP to know the type of attribute requested by the SP. However, the IDP does not know which attribute in its database has been retrieved and sent to the SP. The knowledge of attribute type allows IDP to cross-check if SP has permission to retrieve the listed attributes. For instance, a webshop selling books does not need to know which city the user was born in.

Research Question 1 A. *Can leveled homomorphic encryption provide a scalable and efficient solution to improve the privacy of users at the IDP by preventing the IDP from knowing which user it is authenticating to an SP?*

`PReID` uses a leveled homomorphic encryption based PIR to improve privacy of users. We have analysed the computation and communication cost of incorporating DSH PIR in an eID context. Through our implementations we have been able to analyse the overhead that an eID scheme would incur on incorporating PIR. Retrieving data from a single large database is not efficient in terms of computation time. If PIR is run on smaller segments of the entire database, then the scheme becomes much more efficient and consequently makes the solution scalable. But allowing IDP to partition the database into smaller segments would allow it to profile users. Thus we make a trade-off in terms of the database partition size for efficiency, scalability and limiting IDP's profiling probability. For encrypted attributes, $d = 3$ circuit provides reasonable performance. But the space required to store at the IDP is proportional to the number of attributes as well as the number of SPs chosen by the user. For plain attributes, we suggest to partition the database such that the attributes can be retrieved with $d = 4$ circuit. With the improvements mentioned in Section 4.5, we foresee LHE-based PIR being incorporated in federated eID context with a 3 second overhead for `PIR.Response`. At the user side, we suggest TEE on mobile phones as possible deployment environment to store database indices and to execute `PIR.Query`.

Thus we conclude that leveled homomorphic encryption can provide a scalable and efficient solution to improve user privacy in a federated eID context if we are willing to partition the database and concede limited profiling capability to the IDP.

There are many directions for future work. We have used PIR in eID context, but PIR can be ported to any three-party scenario where the entity querying the server has the public key of the entity which receives the retrieved data. So the first step would be to find applications which will benefit from using PIR. Second, we have focussed on using LHE in our work and limiting the number of communication rounds to 1. Further research could look into the possibility of using other PIR schemes in eID context which have a greater bandwidth complexity but reduce the computation time for retrieval. Third, while storing encrypted attributes, we performed encryption using the public key of SP. This method increases the storage space required at the IDP as the same data needs to be encrypted for multiple SPs. Also the user needs to know which SPs service it will avail. To counter this issue, attributes can be stored using the user's key or another key generated by a third-party for which the IDP does not have the corresponding private key. This method opens up the possibility of using multi-key LHE. Another option is investigate the possibility of using proxy re-encryption along with PIR. Fourth, when a SP requires to know if the user is above a certain age, we send the date of birth attribute. The alternative is to store the age of the user. But this option requires a cumbersome procedure of changing the attribute value every year. A better way of handling derived attributes in a federated scheme needs to be investigated. Fifth, lattice-based cryptography is not conducive for implementation on smart cards. Though we have mentioned the possibility of using TEE in mobile phones, further research is required in this direction. Finally, we also mention an idea that we had considered in our literature survey [21] that needs to be researched further. We looked into the possibility of using block chain to decentralize the role of IDP in eID schemes. In this context, there remains a possibility of using multi-party computation (MPC).

Bibliography

[1] *Too many passwords to remember*, (Accessed on April 12, 2016). [Online]. Available: https://www.theguardian.com/technology/askjack/2010/sep/30/password-management-internet.

[2] *National mobile ID schemes*, (Accessed on April 12, 2016). [Online]. Available: http://www.secureidnews.com/news-item/reid-national-eid-series-europe-leads-global-push-to-eid/.

[3] *Re:ID national eID series: Europe leads global push to eid*, (Accessed on April 12, 2016). [Online]. Available: http://www.secureidnews.com/news-item/reid-national-eid-series-europe-leads-global-push-to-eid/.

[4] *Idensys*, (Accessed on April 12, 2016). [Online]. Available: https://www.idensys.nl/.

[5] *Polymorphic pseudonymization*, version 0.91, programma eID, (Retrieved on April 12, 2016). [Online]. Available: https://www.idensys.nl/fileadmin/bestanden/idensys/documenten/basisdocumentatie/documentatieset/PP_Scheme_091.pdf.

[6] L. T. A. N. Brandão, N. Christin, G. Danezis, and anonymous, "Toward mending two nation-scale brokered identification systems", *PoPETs*, vol. 2015, no. 2, pp. 135–155, 2015.

[7] *Gov.uk verify*, (Accessed on April 12, 2016). [Online]. Available: https://www.gov.uk/government/publications/introducing-govuk-verify.

[8] BSI, "Advanced security mechanisms for machine readable travel documents and eIDAS token - part 2 - protocols for electronic identification, authentication and trust services (eIDAS)", Bundesamt fur Sicherheit in der Informationstechnik, TR-03110-2, version 2.20, 2015.

[9] P. Vullers and G. Alpár, "Efficient selective disclosure on smart cards using idemix", in *IDMAN*, ser. IFIP Advances in Information and Communication Technology, vol. 396, Springer, 2013, pp. 53–67.

[10] *Ashley madison data breach*, (Accessed on April 28, 2016). [Online]. Available: https://en.wikipedia.org/wiki/Ashley_Madison_data_breach.

[11] *7 largest data breaches of 2015*, (Accessed on April 28, 2016). [Online]. Available: http://www.healthcareitnews.com/news/7-largest-data-breaches-2015.

[12] *Cost of a data breach*, (Accessed on January 5, 2016). [Online]. Available: https://www.symantec.com/about/news/resources/press_kits/detail.jsp?pkid=ponemon.

[13] *Target puts data breach costs at $148 million and forecasts profit drop*, (Accessed on January 5, 2016). [Online]. Available: https://www.nytimes.com/2014/08/06/business/target-puts-data-breach-costs-at-148-million.html?_r=0.

[14] *Anthem sued over large data breach by california consumer*, (Accessed on April 28, 2016). [Online]. Available: http://www.bloomberg.com/news/articles/2015-02-06/anthem-sued-in-california-by-consumer-over-massive-data-breach.

[15] *Data breaches now resulting in more lost customers*, (Accessed on January 5, 2016). [Online]. Available: https://blog.digicert.com/cost-data-breaches-2014/.

[16] *22 million affected by OPM hack, officials say*, (Accessed on April 28, 2016). [Online]. Available: http://abcnews.go.com/US/exclusive-25-million-affected-opm-hack-sources/story?id=32332731.

[17] *Turkish citizenship database leak*, (Accessed on April 14, 2016). [Online]. Available: http://www.databreaches.net/turkish-citizenship-database-leak/.

[18] *Ph: Voter database leaked by hackers*, (Accessed on April 14, 2016). [Online]. Available: http://www.databreaches.net/ph-voter-database-leaked-by-hackers/.

[19] *The darkest side of ID theft*, (Accessed on April 14, 2016). [Online]. Available: http://www.nbcnews.com/id/3078488/ns/technology_and_science-tech_and_gadgets/t/darkest-side-id-theft/.

[20] *The holocaust in the netherlands and the rate of jewish survival*, (Accessed on April 14, 2016). [Online]. Available: http://www.yadvashem.org/yv/en/education/languages/dutch/pdf/article_croes.pdf.

[21] K. Shrishak, "Enhancing the privacy of users in eid schemes through cryptography", Literature Survey, Delft University of Technology, Delft, 2016, p. 45.

[22] *What's privacy?*, (Accessed on April 28, 2016). [Online]. Available: http://www.rogerclarke.com/DV/Privacy.html.

[23] B. Zwattendorfer and D. Slamanig, "The austrian eid ecosystem in the public cloud: How to obtain privacy while preserving practicality", *CoRR*, vol. abs/1601.03533, 2016.

[24] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography", in *EUROCRYPT*, ser. Lecture Notes in Computer Science, vol. 1403, Springer, 1998, pp. 127–144.

[25] D. Nuñez, I. Agudo, and J. Lopez, "Integrating openid with proxy re-encryption to enhance privacy in cloud-based identity services", in *CloudCom*, IEEE Computer Society, 2012, pp. 241–248.

[26] ISO15408-2:2005, "Information technology - security techniques - evaluation criteria for it security - part 2: Security functional requirements", International Standard Organization, Tech. Rep., 2005.

[27] Y. Doröz, B. Sunar, and G. Hammouri, "Bandwidth efficient PIR from NTRU", in *Financial Cryptography Workshops*, ser. Lecture Notes in Computer Science, vol. 8438, Springer, 2014, pp. 195–207.

[28] A. López-Alt, E. Tromer, and V. Vaikuntanathan, "On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption", *IACR Cryptology ePrint Archive*, vol. 2013, p. 94, 2013. [Online]. Available: http://eprint.iacr.org/2013/094.

[29] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings", in *EUROCRYPT*, ser. Lecture Notes in Computer Science, vol. 6110, Springer, 2010, pp. 1–23.

[30] X. Yi, R. Paulet, and E. Bertino, *Homomorphic Encryption and Applications*, ser. Springer Briefs in Computer Science. Springer, 2014, ISBN: 978-3-319-12228-1.

[31] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms", in *Foundations of Secure Computation*, R. A. DeMillo, D. P. Dobkin, A. K. Jones, and R. J. Lipton, Eds., Academic Press, 1978, pp. 165–179, ISBN: 0122103505.

[32] C. Gentry, "A fully homomorphic encryption scheme", PhD thesis, Stanford University, 2009.

[33] C. Gentry and S. Halevi, "Implementing gentry's fully-homomorphic encryption scheme", in *EUROCRYPT*, ser. Lecture Notes in Computer Science, vol. 6632, Springer, 2011, pp. 129–148.

[34] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping", in *ITCS*, ACM, 2012, pp. 309–325.

[35] J. Hoffstein, J. Pipher, and J. H. Silverman, "NTRU: A ring-based public key cryptosystem", in *ANTS*, ser. Lecture Notes in Computer Science, vol. 1423, Springer, 1998, pp. 267–288.

[36] D. Stehlé and R. Steinfeld, "Making NTRU as secure as worst-case problems over ideal lattices", in *EUROCRYPT*, ser. Lecture Notes in Computer Science, vol. 6632, Springer, 2011, pp. 27–47.

[37] Y. Doröz, Y. Hu, and B. Sunar, "Homomorphic AES evaluation using NTRU", *IACR Cryptology ePrint Archive*, vol. 2014, p. 39, 2014.

[38] L. D. Martin Albrecht Shi Bai, *A subfield lattice attack on overstretched ntru assumptions: Cryptanalysis of some fhe and graded encoding schemes*, Cryptology ePrint Archive, Report 2016/127, http://eprint.iacr.org/, 2016.

[39] C. Gentry, S. Halevi, and N. P. Smart, "Homomorphic evaluation of the AES circuit", in *CRYPTO*, ser. Lecture Notes in Computer Science, vol. 7417, Springer, 2012, pp. 850–867.

[40] N. Gama and P. Q. Nguyen, "Predicting lattice reduction", in *EUROCRYPT*, ser. Lecture Notes in Computer Science, vol. 4965, Springer, 2008, pp. 31–51.

[41] R. Lindner and C. Peikert, "Better key sizes (and attacks) for lwe-based encryption", in *CT-RSA*, ser. Lecture Notes in Computer Science, vol. 6558, Springer, 2011, pp. 319–339.

[42] D. Coppersmith and A. Shamir, "Lattice attacks on NTRU", in *EUROCRYPT*, ser. Lecture Notes in Computer Science, vol. 1233, Springer, 1997, pp. 52–61.

[43] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval", in *FOCS*, IEEE Computer Society, 1995, pp. 41–50.

[44] Y. Ishai and E. Kushilevitz, "Improved upper bounds on information-theoretic private information retrieval (extended abstract)", in *STOC*, ACM, 1999, pp. 79–88.

[45] B. Chor and N. Gilboa, "Computationally private information retrieval (extended abstract)", in *STOC*, ACM, 1997, pp. 304–313.

[46] E. Kushilevitz and R. Ostrovsky, "Replication is NOT needed: SINGLE database, computationally-private information retrieval", in *FOCS*, IEEE Computer Society, 1997, pp. 364–373.

[47] C. Cachin, S. Micali, and M. Stadler, "Computationally private information retrieval with polylogarithmic communication", in *EUROCRYPT*, ser. Lecture Notes in Computer Science, vol. 1592, Springer, 1999, pp. 402–414.

[48] C. Gentry and Z. Ramzan, "Single-database private information retrieval with constant communication rate", in *ICALP*, ser. Lecture Notes in Computer Science, vol. 3580, Springer, 2005, pp. 803–815.

[49] H. Lipmaa, "An oblivious transfer protocol with log-squared communication", in *ISC*, ser. Lecture Notes in Computer Science, vol. 3650, Springer, 2005, pp. 314–328.

[50] C. A. Melchor and P. Gaborit, "A lattice-based computationally-efficient private information retrieval protocol", *IACR Cryptology ePrint Archive*, vol. 2007, p. 446, 2007.

[51] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption", *IACR Cryptology ePrint Archive*, vol. 2012, p. 144, 2012.

[52] *Ntl: A library for doing number theory.* [Online]. Available: http://www.shoup.net/ntl.

[53] T. Granlund and the GMP development team, *Gnu mp: The GNU Multiple Precision Arithmetic Library*, 6.1.0, http://gmplib.org/, 2016.

[54] N. P. Smart and F. Vercauteren, "Fully homomorphic SIMD operations", *IACR Cryptology ePrint Archive*, vol. 2011, p. 133, 2011.

[55] W. Dai, Y. Doröz, and B. Sunar, "Accelerating SWHE based pirs using gpus", in *Financial Cryptography Workshops*, ser. Lecture Notes in Computer Science, vol. 8976, Springer, 2015, pp. 160–171.

[56] A. Vasudevan, E. Owusu, Z. Zhou, J. Newsome, and J. M. McCune, "Trustworthy execution on mobile devices: What security properties can my mobile platform give me?", in *TRUST*, ser. Lecture Notes in Computer Science, vol. 7344, Springer, 2012, pp. 159–178.

[57] N. C. Dwarakanath and S. D. Galbraith, "Sampling from discrete gaussians for lattice-based cryptography on a constrained device", *Appl. Algebra Eng. Commun. Comput.*, vol. 25, no. 3, pp. 159–180, 2014.